УДК 004.4’2

к.т.н., доц. **Муляр І.В.** (ХмНУ)
к.т.н. **Лєнков Є.С.** (ЦНДІ ЗСУ)
Жовнір С.М. (ХмНУ)
Кушнірук С.Л. (ХмНУ)

АДАПТИВНИЙ МЕТОД ПРОЕКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У статті запропоновано підхід до оптимального проектування і формування багатометодних стратегій пошуку найкращих варіантів у відповідності зі специфікою вирішуваних завдань.

Так як розробка програмного забезпечення набуває великого значення, потрібно проводити ефективне проектування на початкових етапах його життєвого циклу. Це дозволить ефективніше використовувати час, матеріальні та людські ресурси. Саме тому в роботі проведено аналіз напрямків підвищення ефективності процесів прийняття рішень при проектуванні складних програмних систем з можливістю перебудови. Визначені принципи адаптивної алгоритмізації задач оптимального проектування програмних систем з урахуванням динамічного та стохастичного характеру проектованої системи. З метою забезпечення гнучкості і адаптованості алгоритмічної бази оптимального проектування програмних систем пропонується використовувати компонентно-модульний підхід, заснований на структуризації оптимізаційних задач і алгоритмів їх вирішення. Побудовані багаторівневі ітеративні схеми оптимального проектування програмних систем з можливістю координації проектних рішень на різних рівнях декомпозиції. Сформовані математичні моделі оптимального вибору структурі програмної системи та її окремих елементів при проектуванні програмних систем чи перебудові вже існуючих. Запропонований метод формування адаптивного багатометодного середовища пошуку проектних рішень, що забезпечує побудову процедур структурного та параметричного синтезу різних класів за допомогою комплектування інваріантних структурних компонентів. Розроблено структуру і склад бібліотеки модулів оптимального проектування, що характеризуються наявністю груп взаємозамінних компонентів і можливістю формування різних за призначенням та інформаційної наповненості алгоритмів допомогою комбінацій-модулів. Запропоновано організацію бази знань у вигляді сукупності системи фреймів, що містять відомості про структуру алгоритмічної бази та характеристиках оптимізаційних задач, а також набору правил комплексування модулів.

Ключові слова: програмна система, оптимальне проектування, пошук проектних рішень, база знань.

Вступ. В теперішніх умовах багато уваги приділяється проблемам, що пов’язані з проектуванням складних програмних систем. При цьому значний комплекс проектних завдань пов'язаний не лише з створенням нових програмних комплексів, але і з необхідністю структурної та параметричної перебудови діючих систем з метою їх реформування та модернізації. Динамічні зміни зовнішніх умов, потребують постійного технічного та алгоритмічного удосконалення. Це призводить до необхідності розробки математичного та програмного забезпечення для комплексного рішення задач аналізу та синтезу складних програмних систем з урахуванням ієрархічності та можливості перебудови структури програмного забезпечення.

Процес структурного та параметричного синтезу програмних систем ускладнюється високою динамікою, нестабільністю та стохастичністю вимог до програмного забезпечення, що ускладнює використання аналітичних моделей для прийняття проектних рішень [1]. Це призводить до необхідності застосування алгоритмічних оптимізаційних моделей, в яких відсутні явні аналітичні формулювання критеріїв оптимальності та обмежень, а наявна лише можливість визначення їх значень для кожного із варіантів з застосуванням різних моделюючих процедур. Складність оцінки властивостей таких моделей обмежує можливості використання стандартних оптимізаційних процедур, що в кінцевому результаті знижує ефективність процесу оптимального проектування [2, 3]. Крім того, до особливостей задач оптимального проектування складних програмних систем можна віднести різновиди постанов, високу розмірність, різноманітність техніко-економічних вимог до основних характеристик,

значну трудомісткість етапів моделювання та аналізу, наявність різновиду кореляційних зав’язків між параметрами, які досить важко врахувати в рамках стандартного математичного забезпечення САПР [4].

Рішенням вказаної проблеми може бути досягнуто при використанні адаптивного підходу до проектування програмних систем. Даний підхід передбачає побудову комплексу алгоритмів оптимізації які забезпечують поєднання процесу більш повної формалізації задачі з її рішенням, та їх поєднання з процедурами багатоваріантного моделювання в інтелектуальну адаптивне середовище з можливістю його динамічного налаштування на різні класи вирішуваних задач оптимального проектування. При цьому, важливою вимогою до розроблюваного математичного забезпечення являється можливість ефективного рішення задач що описані складними алгоритмічними моделями.

Постановка задачі. Таким чином, актуальність дослідження визначається необхідністю розробки теоретичних основ, математичного та програмного забезпечення для рішення задач пошуку оптимальних варіантів складних програмних систем при їх реформуванні та модернізації з можливістю врахування динамічних та стохастичних аспектів роботи.

Для цього необхідно розробити метод, моделі та алгоритми оптимального проектування програмних систем з можливістю перебудови, та створити на цій основі адаптивне середовища пошуку раціональних проектних рішень в умовах змін вимог до проектованої системи.

Основна частина. Структура програмної системи залежить від форм та методів організація процесів пов’язаних з обробкою та збереження даних, від кількості даних, їх спеціалізації, кооперації та комбінування. Глибина розділення процесу обробки даних на його складові частини, та ступінь кооперації визначають види та призначення компонентів програми.

Першим етапом розробки нової програмної системи або удосконалення вже існуючої, є комплексний всебічний аналіз всіх характеристик яким система повинна відповідати.

При побудові імітаційних моделей процесів функціонування складних програмних систем можна виділити наступні основні підходи: безперервно-детермінований (диференційні рівняння); дискретно детерміновані (кінцеві автомати); дискретно-стохастичні (ймовірнісні автомати); безперервно стохастичні (системи масового обслуговування); узагальнений або універсальний (агрегатні системи). Перераховані підходи дозволяють формалізувати достатньо широкий клас об’єктів [5].

Ефективність процесів оптимального проектування виробничих систем однаковою мірою визначається як властивостями використовуваних алгоритмічних процедур, так і організацією відповідних програмних систем оптимізації як інтегрованого інструмента аналізу властивостей задачі і управління процесом його вирішення. З ускладненням завдань і посиленням вимог до обґрунтованості прийнятих рішень необхідний всебічний, комплексний аналіз об'єкта проектування, врахування взаємозв'язку багатьох чинників у їх динаміці, забезпечення можливості гнучкого та оперативного управління оптимізаційним процесом [6, 7]. Практична реалізація такого підходу призводить до побудови інтегрованих середовищ оптимізації, в яких поряд з процедурами, які забезпечують вирішення самої задачі оптимального проектування, наявні підсистеми управління базою даних, комплекс "сервісних" програм, модулі формування та коригування моделі, блоки аналізу та обробки результатів оптимізаційних розрахунків та ін. Сучасні діалогові системи оптимального проектування об'єднують в гнучкому інтерактивному режимі формальні і неформальні елементи оптимізаційного процесу в єдину людино-машинну технологію прийняття раціональних проектних рішень [8, 9].

Рішення задач оптимального проектування програмних систем з можливістю перебудови на різних ієрархічних рівнях передбачає виділення типових завдань структурно-параметричної перебудови програми та формування комплексу математичних моделей структурного і параметричного синтезу як програмної системи в цілому, так і її окремих елементів [10]. Особливістю формованого комплексу моделей є використання імітаційного моделювання в контурі прийняття рішень з метою врахування динаміки функціонування і

стохастичного характеру виробничою системи. Побудуємо оптимізаційну модель вибору програмних маршрутів обробки даних.

Розглянемо множину об'єктів програмної системи r_j, $j=\overline{1,J}$, які можуть проходити обробку у конкретній функції загального типу. Необхідно вибрати мінімальну групу R елементів множини r_j, якими можна найбільш ефективно завантажити функції загального типу. З цією метою введемо альтернативні змінні:

$$x_j=\begin{cases}1, \textit{ якщо } j-\textit{й об'єкт програмної системи входить до групи } R\\ 0 \textit{ в протилежному випадку}\end{cases}.$$

Ефективність завантаження системи при обробці j-го об'єкта програмної системи будемо розглядати наступним чином:

$$c_j=\begin{cases}1 \textit{ якщо } i-\textit{та операція, яка може виконуватись компонентом,}\\ \textit{використовується для обробки } j-\textit{го об'єкта програмної}\\ \textit{системи, } i=\overline{1,I},\ j=\overline{1,J}\\ 0 \textit{ в протилежному випадку.}\end{cases}.$$

Тут I – загальна кількість технологічних операцій, які можуть виконуватись функція загального типу.

Ефективність використання таких функцій досягається в тому випадку, якщо при її обробці призначених для них об'єктів кожна i-та програмна операція буде виконуватись не менше одного разу, тобто:

$$\sum_{j=1}^{J} c_{ij}x_j \geq 1, \tag{1}$$

Вище вказувалось про формування мінімальної групи елементів R. У цьому випадку оптимізаційна модель буде мати вигляд:

$$\begin{gathered}\sum_{j=1}^{J} x_j \to \min\\ \sum_{j=1}^{J} c_{ij}x_j \geq 1\\ x_j \in \{0;1\}\ i=\overline{1,I},\ j=\overline{1,J}\end{gathered}. \tag{2}$$

Оптимізаційна модель (2) формалізована як задача про мінімальне покриття.

З першого етапу проектування структури програмного процесу системи з функціями загального типу виходить необхідність вибору раціонального варіанту концентрації програмних операцій та переналаштування алгоритму при переході від однієї групи технологічних операцій до другої. Ці задачі пов'язані між собою та визначають доцільність входження i-ї програмної операції в деяку k-ту ($k=\overline{1,K}$) групу однорідних по програмному способі обробки даних. Тому їх рішення має сенс визначати на основі єдиної математичної оптимізаційної моделі.

Нехай для обробки j-го об'єкта обробки необхідно виконати сукупність технологічних операцій $v1_j,...,v_{ij},...,v_{Ij}$. Час що необхідний для виконання кожної програмної операції рівний τ_{ij}. Визначено число груп програмних операцій k, які однорідні по призначенню в процесі послідовного використання функцій загального типу для обробки об'єктів системи r_j. Варіанти що передували програмним операціям для кожного об'єкта обробки задані наступними послідовностями

$$v_{ij_m} \ \pi_m \ v_{lj}\ ,\ \{i,l\} \in \overline{1,I}\ \ ,\ l \neq i,\ m = \overline{1,M},$$

де π - знак передування, m вказує на номер варіанту програмного процесу обробки даного об'єкту програмної системи.

Необхідно знобити такий розподіл сукупностей технологічних операцій між групами, що:

1) задовільнялись умови передування;

2) сумарна трудомісткість виконання програмних операцій для всіх об'єктів програмної системи була мінімальною;

3) сумарний час переходу від алгоритму до алгоритма було мінімальним;

4) виконувались умови по обмеженню кількості технологічних оперцій в групі;

5) задовільнялось обмеження по часу виконання i-ї програмної операції, $i = \overline{1,L}$ в кожній групі по виду алгоритму (тобто обмеження по тривалості виконання алгоритму).

Для побудови оптимізаційної моделі введемо альтернативні змінні:

$$x_{ijk} = \begin{cases} 1 \text{ якщо } i-\text{та операція, що використовується при} \\ \quad \text{обробці } j-\text{го об'єкта програмної системи} \\ \quad \text{в } k-\text{ту групу;} \\ 0 \text{ в протилежному випадку.} \end{cases}$$

$$i = \overline{1,I},\ j = \overline{1,J},\ k = \overline{1,K}$$

$$y_{jm_j} = \begin{cases} 1 \text{ якщо при обробці } j-q \text{ даних вибирається} \\ \quad m-\text{й варіант передування операцій} \\ 0 \text{ в протилежному випадку.} \end{cases}$$

При цьому значення змінних x_{ijk} залежать від значень y_{jm_j}, так як варіанти передувань визначають розподіл технологічних операцій між групами. Цей факт буде позначатись $x_{ijk}(y_{jm_j})$.

Тепер запишемо у формалізованому вигляді викладенні вище вимоги до розподілу програмних операцій.

Умови передування програмних операцій враховуються за допомогою обмежень

$$\sum_{k=1}^{K} x_{ijk}(y_{jm_j}) \le \sum_{k=1}^{K_1} x_{ljk}(y_{jm_j}),\ k_1 = 1,2,...,K \quad . \qquad (3)$$
$$j = \overline{1,J},\ m_j = \overline{1,m_j},\ \{j,l\} \in 1,I,\ i \neq l$$

Кожна програмна операція при обробці j-го об'єкту повинна відноситись лише до однієї з груп, тому:

$$\sum_{k=1}^{K} x_{ijk} = 1,\ i = \overline{1,I},\ j = \overline{1,j}. \qquad (4)$$

При розробці моделі, також необхідно враховувати, що при обробці j-го об'єкту програмної системи, вибирається лише один варіант передування (одни програмний процес):

$$\sum_{m_j=1}^{M_j} y_{jm_j} = 1,\ j = \overline{1,j}. \qquad (5)$$

Сумарний час виконання i-ї операції в k-й групі обмежено часом роботи алгоритму. Це можна записати у вигляді наступного обмеження:

$$\sum_{j=1}^{J} x_{ijk} \sum_{m_j=1}^{M_j} \tau_{ij}(y_{jm_j}) \le B_j, \ \ i=\overline{1,I}, \ k=\overline{1,K}, \tag{6}$$

де B_i -максимальний час використання і-го алгоритму в одній групі.

При оптимальному проектуванні складних програмних систем важливу роль відіграє побудова проблемно-адаптивних процедур, що дозволяють формувати оптимізаційні моделі і вибирати найбільш ефективні обчислювальні схеми параметричного і структурного синтезу програмних систем в залежності від специфіки вирішуваних завдань оптимального проектування [11].

Структурна організація інтелектуального багатометодного середовища представлена на рис. 1.

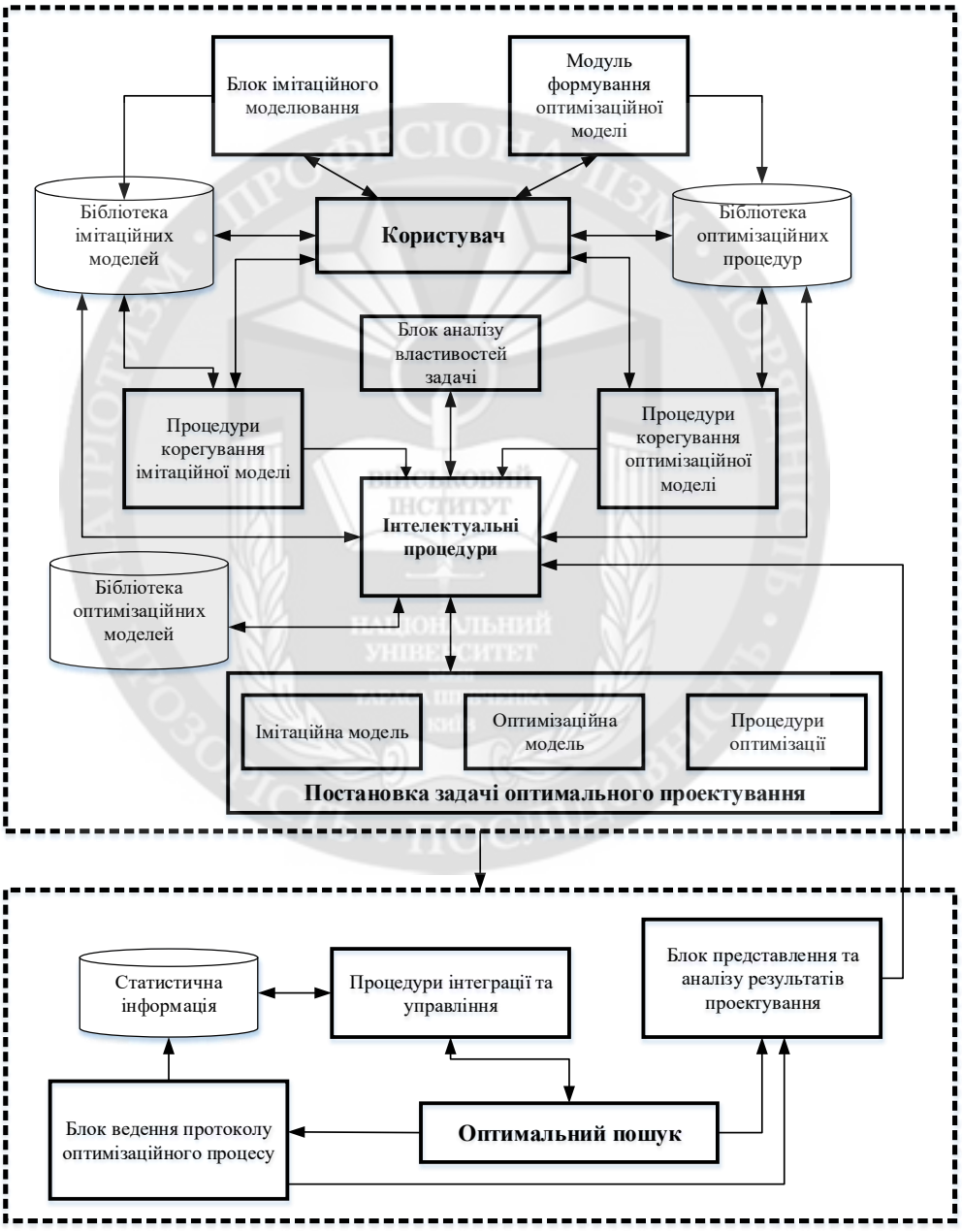

Рис. 1. Організація багатометодного середовища оптимального проектування програмної системи

Практична реалізація принципу проблемної адаптації передбачає організацію алгоритмічного та програмного забезпечення аналізу та синтезу складних програмних систем у вигляді адаптивного інтелектуального багатометодного середовища оптимального проектування [5]. При цьому інтелектуальне пошукове середовище визначається як сукупність алгоритмічних процедур та програмних засобів, що забезпечують налаштування алгоритмічної бази оптимального проектування на різні класи вирішуваних задач структурного та параметричного синтезу

Бібліотека оптимізаційних моделей оптимального проектування програмних систем включає математичні моделі для вирішення задач параметричного і структурного синтезу різних класів. Узагальнена структура бібліотеки представлена на рис. 2.

Рис. 2. Загальна структура бібліотеки оптимізаційних моделей

Для адаптивного налаштування алгоритмічної бази відповідно до особливостей вирішуваних завдань оптимального проектування пропонується використовувати проблемно-адаптивні процедури, що забезпечують підтримку комплектування модулів оптимального проектування і формування багатометодних стратегій організації пошуку найкращих варіантів на підставі динамічної апріорної і апостеріорної оцінки властивостей розв'язуваної задачі [12].

Структурна схема інтелектуальної підсистеми представлена на рис. 3.

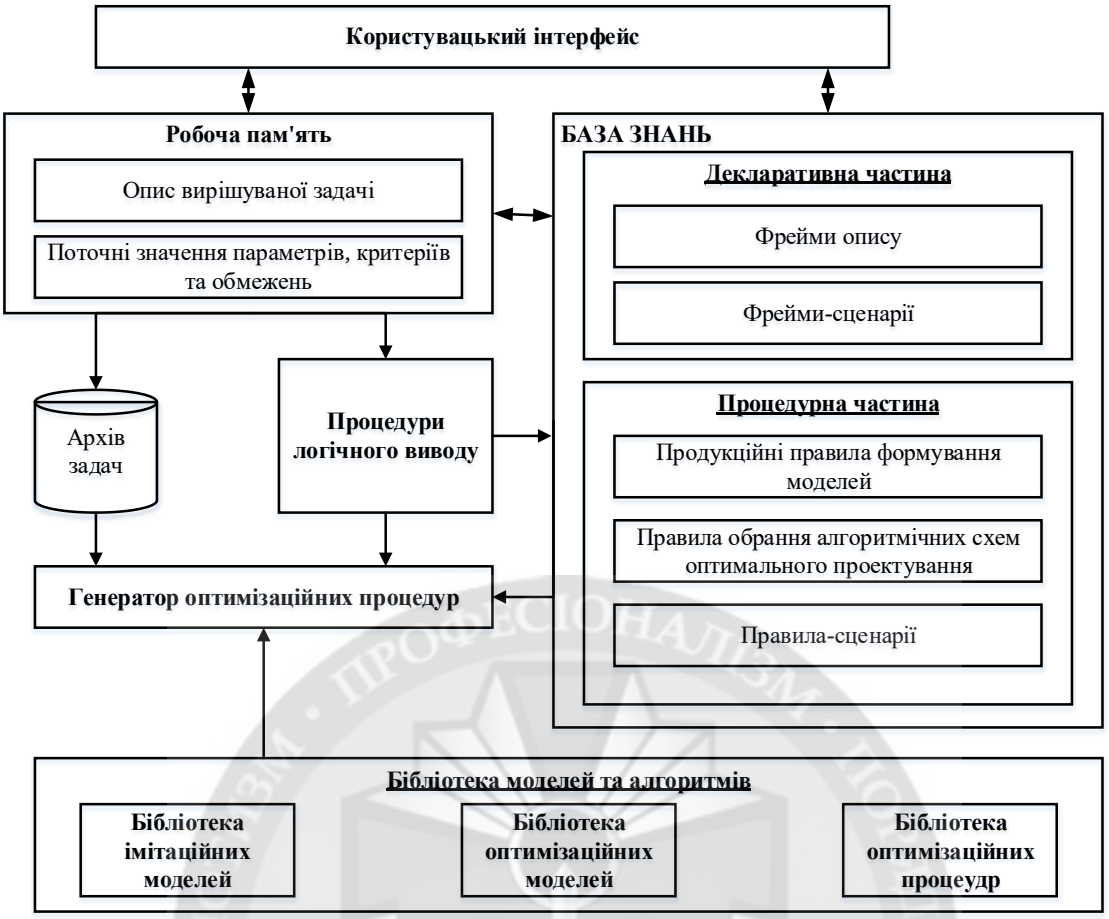

Рис. 3. Структурна схема інтелектуальної підсистеми

База знань структурована і розділена на декларативну і процедурну частину. Декларативні знання представлені у вигляді системи фреймів. Кожен фрейм має стандартну для інтелектуальних систем структуру:

$$< nf, pr, (ns_1, zs_1, np_1), ..., (ns_m, zs_m, np_m) >,$$

де nf - ім'я фрейму; pr - умова застосування фрейму; ns - ім'я слоту; zs - значення слоту; np - ім'я підключеної процедури; m - кількість слотів фрейму. При цьому виділяються наступні типи фреймів:

1) фрейми-опису, що містять відомості про структуру алгоритмічної бази та характеристики вирішуваних задач;

2) фрейми-сценарії, що відображають послідовності дій по формуванню різних багатометодних стратегій організаційного процесу.

До першої групи фреймів можна віднести наступні:

3) Описи різних класів алгоритмів, представлені у вигляді: *<Фрейм алгоритму>: = <ім'я алгоритму> | <клас алгоритму> | <опис> | <умови застосування> | <рекомендовані параметри> | <посилання на схеми складання модулів, що належать даному класу алгоритмів>.*

4) Опису модулів і їх взаємозв'язків, представлені у вигляді: *<Фрейм модуля>: = <ім'я модуля> | <приналежність до класу модулів> | <призначення> | <реалізована стратегія> | <вхідні параметри> | <вихідні параметри> | <посилання на алгоритм> | <посилання на схеми складання> | <Викликаються модулі>.*

5) Описи схем комплексування модулів. Відповідні фрейми мають вигляд: *<Фрейм схеми складання>: = <ідентифікатор схеми> | <приналежність до класу алгоритмів> | <призначення> | <Послідовність модулів> | <рекомендовані параметри> | <приєднана процедурам>.*

При цьому приєднана процедура, що запускається при активізації даного фрейма, здійснює синтез відповідних програмних компонент.

6) Описи оптимізаційних моделей, представлені у вигляді: *<Фрейм оптимізаційної моделі>: = <ім'я моделі> | <клас моделі> | <опис> | <рекомендовані параметри>.*

7) Описи імітаційних моделей, представлені у вигляді: *<Фрейм імітаційної моделі>:: = <ім'я моделі>] Належність до класу моделей> | <опис> | <вхідні параметри> | <вихідні параметри> | <умови застосування>.*

Другий клас фреймів, фрейми-сценарії, можна інтерпретувати як функціональні ієрархії прийняття рішень в стереотипних проектних ситуаціях. Фрейми даного класу узагальнено можуть бути представлені у вигляді:

<Фрейм-сценарій> :: = <умови застосовності фрейма> | <Найменування сценарію> | <умова 1> <дія 1> <приєднана процедура 1> | ... | <умова m> <дія m> <приєднана процедура m>.

Умова застосування фрейму являє собою n-місний предикат. Якщо на деякому наборі предметних змінних предикат приймає значення "Істина", відбувається активізація даного фрейма-сценарію. При цьому фрейм-сценарій містить послідовність дій з формування різних багатометодних стратегій організації оптимізаційного процесу. З кожною дією зв'язується процедура що реалізує його, та є фрагментом керуючої програми, а також умову (розпізнавач ситуації), при виконанні якого здійснюється реалізація даної дії.

Процедурна частина бази знань містить набір правил синтезу алгоритмічних схем, що представляють собою продуційні відношення на підмножинах множини W. Кожний j-й запит має наступний вигляд:

$$(j),\ якщо\ \ antec_j,\ то\ canseq_j\ \ \ CF^p = cf_j^p,$$

де j – індекс запиту або даних; $antec_j$ - антецедент j-го запиту або даних; $conseq_j$ - консеквент j-го запиту або даних; $CF^p \in [0;1]$ - ступінь істинності правила, що визначається на етапі формування бази знань та така що характеризує суб'єктивну впевненість розробника в істинності даного правила. Антецедент $antec_j$ в загальному випадку є кон'юнкцією елементарних посилань a_{jq}:

$$antec_j = a_{j1}\ И\ a_{j2}\ И ...\ И\ a_{js_j},$$

де s_j - загальна кількість елементарних посилань в j-му запиті або даних. Консеквентна $conseq_j$ представляє собою елементарну послідовність b_j, що виконується у випадку істинності антецедента $antec_j$ та полягає у формуванні чергового варіанту.

Елементарне посилання a_{jq} та послідовність b_j є диз'юнкцією елементарних факторів предметної області f_{jq}:

$$w_{i_{jq}} = v_{jq}^1\ \ АБО\ \ w_{i_{jq}} = v_{jq}^2\ \ АБО...,$$

де jq – індекс об'єкта, що входить в q-те елементарне посилання j-го запиту чи даних; v_{jq}^k - значення даного об'єкта. При цьому елементарне посилання одного запиту може бути наслідком іншого запиту, що породжує відповідне дерево логічного виводу. Істинність посилання визначається на основі співставлення заданих поточних значень об'єктів, що входять до елементарних посилань, та значень що записані в запиті або даних.

Правила в базі знань структуруються наступним чином:

8) правила формування моделей (оптимізаційних та імітаційних);

9) правила вибору алгоритмічних схем оптимального проектування;

10) правила-сценарії, на основі яких відбувається послідовна схема алгоритмічних схем при оптимальному проектуванні.

Визначення імітаційної моделі здійснюється залежно від об'єкта проектування [9]. При цьому з бібліотеки імітаційних моделей вибирається модель, відповідна даному об'єкту. Вибір оптимізаційної моделі проводиться залежно від класу розв'язуваної задачі. Відповідна оптимізаційна модель також може бути обрана з бібліотеки. При необхідності вибрані моделі можуть бути скориговані користувачем. Алгоритмічні схеми оптимального проектування визначаються на підставі оцінки характеристик вирішуваних завдань.

Вся поточна інформація зберігається в робочій пам'яті, структурно розділеної на кілька блоків. У першому з них містяться загальні відомості про завдання, в інших - дані, необхідні для вирішення конкретних підзадач. Структуризація робочої пам'яті дозволяє скоротити час рішення за рахунок використання на кожному етапі синтезу тільки необхідної інформації та зменшення витрат на зіставлення. Перехід до нової задачі пов'язаний з очищенням робочої пам'яті і приміщенням в неї нових даних.

Висновки. Розроблена структура і склад бібліотеки модулів оптимального проектування характеризуються наявністю груп взаємозамінних компонентів і можливістю формування різних за призначенням та інформаційної наповненості алгоритмів допомогою комбінацій-модулів. Бібліотека включає модулі двох класів: інваріантні модулі, що реалізують процедури скалярною безумовної оптимізації; зовнішні модулі, що підключаються до інваріантного алгоритмічного ядра в залежності від специфіки розв'язуваної оптимізаційної задачі (розмірності, типів критеріїв і обмежень і т.д.). Запропоновано організацію бази знань у вигляді сукупності системи фреймів, що містять відомості про структуру алгоритмічної бази та характеристиках оптимізаційних задач, а також набору правил комплексування модулів. Використання розглянутого адаптивного підходу до формування алгоритмічної бази оптимального проектування забезпечує підтримку прийняття рішень при формуванні оптимізаційних моделей і обчислювальних процедур для проектування виробничих систем різних класів. Подальший розвиток інтелектуальної підсистеми передбачає розробку засобів автоматичної ідентифікації властивостей завдань за допомогою дослідження характеристик оптимізаційних моделей перед початком процесу проектування.

ЛІТЕРАТУРА:

1. Martin Fowler. Patterns of Enterprise Application Architecture / Martin Fowler – 1st Edition - Addison-Wesley Profession, 2012, 558 с.

2. Вирт Н. Алгоритмы и структуры данных / Н. Вирт. - М.: ДМК Пресс, 2010. – 272 с.

3. Васильков, А.В. Информационные системы и их безопасность: Учебное пособие / А.В. Васильков, А.А. Васильков, И.А. Васильков. – М.: Форум, 2013. – 528 с.

4. Муляр І.В. Метод вибору шаблонів проектування для вирішення проблем структурної організації коду програмного забезпечення комп’ютерних систем / І.В. Гурман, В.В. Гнатюк, Б.Г. Жиров // Збірник наукових праць Військового інституту Київського національного університету імені Тараса Шевченка. – К.: ВІКНУ, 2015. – Вип. № 50. – С.211-216.

5. Муляр І.В. Метод визначення шаблону проектування вихідного коду програмного забезпечення / Муляр І.В., Гнатюк В.В., Солодєєва Л.В.// Збірник наукових праць Військового інституту Київського національного університету імені Тараса Шевченка. – К.: ВІКНУ, 2015. – Вип. № 49. – С.195-201

6. Martin Fowler. Refactoring. Architecture / Martin Fowler – [s.l.]: Addison-Wesley, 2018, 464 с.

7. Роберт Мартин Гибкая разработка программ на Java и C++. Принципы, паттерны и методики. /Роберт С. Мартин, Джеймс Ньюкирк, Роберт Косс – Изд-во: Диалектика-Вильямс, 2016. – 704 с.

8. Ахо А. Структуры данных и алгоритмы / А. Ахо, Д. Хопкрофт, Д. Ульман / - М.: Вильяме, 2009. - 400 с.

9. Eric Evans, Domain-Driven Design Reference: Definitions and Pattern Summaries / Eric Evans – 1 st Edition - Dog Ear Publishing, 2014, 88 с.

10. Мезенцев К.Н. Автоматизированные информационные системы: Учебник для студентов учреждений среднего проф. образования / К.Н. Мезенцев. - М.: ИЦ Академия, 2013. - 176 с.

11. Scott Millett, Patterns, Principles, and Practices of Domain-Driven Design / Scott Millett – 1st Edition – Wrox, 2015, 792 с.

12. Федорова Г.Н. Информационные системы: Учебник для студ. учреждений сред. проф. образования / Г.Н. Федорова. - М.: ИЦ Академия, 2013. - 208 с.

REFERENCES:

1. Martin Fowler (2012), Patterns of Enterprise Application Architecture, 1st Edition, Addison-Wesley Profession.

2. Virt N. (2010), ”Algoritmy i struktury dannyah” [Algorithms and data structures], DMK Press, Moscow, 272 p.

3. Vasilkov A.V., Vasilkov A.A. and Vasilkov I.A. (2013)” Informacionnye sistemy i ih bezopasnost” [Information systems and their security], Forum, Moscow 528 p.

4. Muliar I.V.,Gurman I.V., Ghnatjuk V.V. and Zhyrov B.G.(2015), ”Method of selecting design patterns for solving the problems of the structural organization of the software code of computer systems” [Method viboru templates in the project for the problem of structural problems of the program software of computer systems], Zbirnyk naukovyh prac' Vijs'kovogo instytutu Kyi'vs'kogo nacional'nogo universytetu imeni Tarasa Shevchenka, No. 50, pp. 211-216.

5. Muliar I.V., Ghnatjuk V.V. and Solodjejeva L.V (2015), ” Metod vyznachennja shablonu proektuvannja vykhidnogho kodu proghramnogho zabezpechennja.” [Method for determining the template for designing the source code of the software], Zbirnyk naukovyh prac' Vijs'kovogo instytutu Kyi'vs'kogo nacional'nogo universytetu imeni Tarasa Shevchenka, No. 49, pp. 195-201.

6. Fowler, M. (2018). Refactoring. [s.l.]: Addison-Wesley.

7. Martin, Robert, Nyukirk Dzheyms and KossRobert (2016), ”Gibkaya razrabotka programm na Java i C. Printsipyi, patternyi i metodiki.” [Flexible development of Java and C ++ programs. Principles, Patterns and Techniques], Dialektika-Vilyams, 704p.

8. Aho A., Hopkroft D.and Ulman D. (2009),” Struktury dannyh i algoritmy” [Data Structures and Algorithms], Vilyame, Moskow, 400 p

9. Eric Evans (2014), Domain-Driven Design Reference: Definitions and Pattern Summaries,1 st Edition, Dog Ear Publishing

10. Mezencev K.N. (2013), ”Avtomatizirovannye informacionnye sistemy” [Automated Information Systems], Akademiya, Moskow 208 p.

11. Scott Millett (2015), Patterns, Principles, and Practices of Domain-Driven Design– 1st Edition – Wrox, 792 c.

12. Fedorova G.N. (2013)” Informacionnye sistemy” [Information Systems], Akademiya, Moskow 176 p.

к.т.н., доц. Муляр И.В., к.т.н. Ленков Е.С., Жовнир С.М., Кушнирук С.Л.

АДАПТИВНЫЙ МЕТОД ПРОЕКТИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

В статье предложен подход к оптимального проектирования и формирования многометодных стратегий поиска лучших вариантов в соответствии со спецификой решаемых задач. Так как разработка программного обеспечения приобретает большое значение, нужно проводить эффективное проектирования на начальных этапах его жизненного цикла. Это позволит более эффективно использовать время, материальные и человеческие ресурсы. Именно поэтому в работе проведен анализ направлений повышения эффективности процессов принятия решений при проектировании сложных программных систем с возможностью перестройки. Определены принципы адаптивной алгоритмизации задач оптимального проектирования программных систем с учетом динамического и стохастического характера проектируемой системы. С целью обеспечения гибкости и адаптируемости алгоритмической базы оптимального проектирования программных систем предлагается использовать компонентно-модульный подход, основанный на структуризации оптимизационных задач и алгоритмов их решения. Построенные многоуровневые итеративные схемы оптимального проектирования программных систем с возможностью координации проектных решений на разных уровнях декомпозиции. Сформированы математические модели оптимального выбора структуре программной системы и ее отдельных элементов при проектировании программных систем или перестройке уже существующих. Предложенный метод формирования адаптивной многометодной среды поиска проектных решений, обеспечивает построение процедур структурного и параметрического синтеза различных классов с помощью комплектования инвариантных структурных компонентов. Разработана структура и состав библиотеки

модулей оптимального проектирования, характеризуются наличием групп взаимозаменяемых компонентов и возможностью формирования различных по назначению и информационной наполненности алгоритмов помощью комбинаций-модулей. Предложено организацию базы знаний в виде совокупности системы фреймов, содержащих сведения о структуре алгоритмической базы и характеристиках оптимизационных задач, а также набора правил комплексирования модулей.

Ключевые слова: программная система, оптимальное проектирование, поиск проектных решений, база знаний.

Ph.D Muliar I.V., Ph.D. Lenkov E.S., Zhovnir S.M., Kushniruk S.V.

METHOD OF ADAPTIVE MULTILEVEL CODING

The main purpose of this work is to develop a method of optimization at transmission of information through communication channels in their noisiness. For the efficient implementation of data transmission in single- and multi-user systems it is needed an adaptation of the scheme transmission to changing conditions of the channel. This project examines the problem of optimizing the parameters of many frequency channels communication in order to minimize the transmitter power required to achieve a certain speed of transmission. Practical implementation of this method leads to the necessity of useing discrete transmission speeds. This leads to the fact that the parameters of the actual data transfer scheme differ from the ideal solution of an optimization task, which leads to some increase in the required power of transmittor. Thus, one can expect increasing the efficiency of the adaptive system in case of using a set of encoding / modulation schemes with a small step speed that can be implemented on the basis of the concept of multilevel coding. Here was presented a new adaptive transfer method for single- and multiplayer system. In the case of a single-user system was used multilevel encoding, which made it possible to improve the accuracy of speed selection encoding / modulation for individual subchannels. In addition, the proposed method of evaluation a bandwidth (as well as spectral efficiency) of adaptive systems which can function in a channel conditions with random independent transmission coefficients, which allows to evaluate characteristics of the system in the area of small ratios of signal / noise. As for the multi-user system case, an adaptive method of power distribution, transmission speed and code division of the channel was proposed, which allowed get an energy gain of up to 3 dB compared to a similar system based on frequency division The method of compression of the generated information service of adaptive system is offered here as well. There is a theoretical lower limit for transmitter power in adaptive system that uses this algorithm. Behavior of the proposed adaptive system was analyzed with the temporary stochastic changes of it state. Also it were presented the results of simulation modeling that characterize the behavior of the proposed method of transmission in conditions of the local cable channel.

Keywords: multifrequency system, data transmission channel, adaptive separation.